GenLeNa: Sistema para la construcción de Aplicaciones de Generación de Lenguaje Natural

Gloria Johanna Chala T.
Analista de sistemas, GC2 Carvajal S.A. Graduada de la Universidad Javeriana Cali, 2006, gjchala@gmail.com

Rafael Armando Jordán O.
Profesor en el Departamento de Ingeniería de la Computación, Universidad Javeriana Cali, rjordan@puj.edu.co

Diego Luis Linares
(Ph. D.) en formas e inteligencia artificial. Universidad Politécnica de Valencia, Coordinador de Investigación U. Javeriana Cali, dlinares@puj.edu.co

Fecha de recepción: 20-10-2006 Fecha de selección: 10-05-2007 Fecha de aceptación: 12-02-2007

ABSTRACT

In this article the proposal is made for the division of the process of construction of natural language generation (NLG) systems into two stages: content planning (CP), which is dependent on the mastery of the application to be developed, and document structuring (DS). This division allows people who are not expert in NLG to develop natural language generation systems, concentrating on building abstract representations of the information to be communicated (called messages). Specific architecture for the DS stage is also presented. This enables NLG researchers to work ortogonally on specific techniques and methodologies for the conversion of messages into text which is grammatically and syntactically correct.

KEYWORDS

Natural Language Generation (NLG), Content Planning (CPP), Document Structuring (DEP), Rhetorical Structure Theory (RST).

RESUMEN

En este artículo se propone la división del proceso de construcción de sistemas de Generación de Lenguajes Natural (GLN) en dos etapas: *planificación del contenido (EPC)*, que es dependiente del dominio de la aplicación a desarrollar, y *estructuración del documento (EED)*. Esta división permite que personas no expertas en GLN puedan desarrollar sistemas de generación de lenguajes natural enfocándose en construir representaciones abstractas de la información que se desea comunicar (denominadas mensajes). Adicionalmente se

presenta una arquitectura específica para la etapa EED que permite a investigadores en GLN trabajar ortogonalmente en técnicas y metodologías específicas para la transformación de los mensajes en texto gramatical y sintácticamente correcto.

PALABRAS CLAVE

Generación de Lenguaje Natural (GLN), Planificación de Contenido (EPC), Estructuración del Documento (EED), Teoría de Estructura Retórica (TER).

Clasificación Colciencias: A

1. INTRODUCCIÓN

Los *sistemas de diálogo* surgen como aplicaciones informáticas que ofrecen un servicio a los usuarios mediante una interacción que debe ser lo más cómoda posible,[15] tratando de imitar la capacidad que tiene el ser humano para hablar y entender. La arquitectura más usada se compone de cinco módulos (ver Figura 1): reconocimiento automático del habla, procesamiento del lenguaje natural, gestión del diálogo, GLN y síntesis del habla; además el sistema se compone de una memoria donde se almacena la información obtenida en el proceso de diálogo y una base de datos de la que el sistema obtiene la información que dará a conocer al usuario.

Un sistema GLN tiene como propósito, a partir de una representación no lingüística de la información, producir texto entendible y gramaticalmente bien escrito para determinado lenguaje (español, inglés, francés, etc.).[9] Estos sistemas son complejos de construir puesto que no sólo se debe decidir qué información comunicar sino cómo transformarla, de modo que se alcance la meta de comunicación deseada. Adicionalmente, dado que la mayoría de sistemas de GLN han sido construidos para dominios específicos (e.g. fumadores, reportes climatológicos, etc.), la forma de expresar las ideas es dependiente de cada uno, y los conceptos, entidades, relaciones, estructuras y métodos usados para realizar la generación del lenguaje en cada sistema en particular no pueden ser fácilmente reutilizados para la construcción de nuevos sistemas GLN.

Existen cuatro métodos de generación o tipos de sistemas: Los *sistemas cerrados (canned)*, fáciles de construir pero poco flexibles en la generación automática de frases, debido a que estas son predefinidas. Los *sistemas basados en plantillas (templates)*. En estos sistemas parte de la plantilla está predefinida y aquella que no lo está se reemplaza posteriormente por

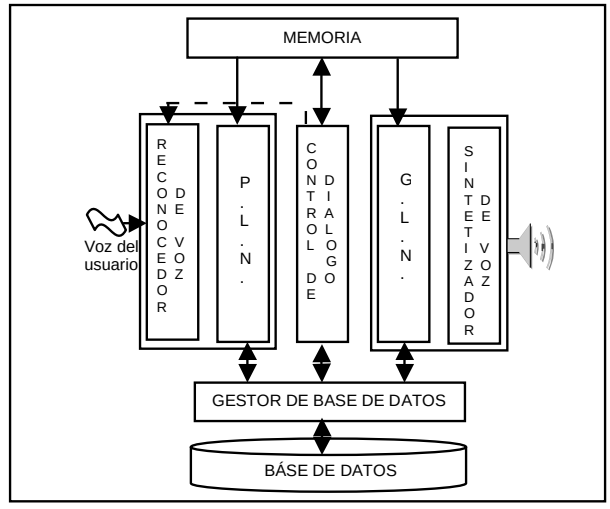

Figura 1. Módulos de un sistema de diálogo.

frases o palabras. Son útiles para dominios no muy grandes y sujetos a pocos cambios. Los *sistemas basados en patrones (phrase)*, que son estructuras que se relacionan entre sí para construir frases (por ejemplo, una oración se puede componer con tres patrones: sujeto, verbo y predicado). Pueden ser bastante potentes y robustos, pero son difíciles de construir debido a la complejidad que existe para relacionar correctamente los patrones que constituyen una frase. Los *sistemas basados en características (feature)*. Una característica da información específica (como género, número, etc.) acerca de las palabras o frases que se generarán. Estos sistemas brindan el mayor nivel de generalidad, pero, al igual que en los sistemas basados en patrones, es difícil mantener las relaciones entre las características, pero adicionalmente el problema principal es seleccionar adecuadamente la combinación de las características que le den el sentido deseado a la frase que se está generando.

Ehud Reiter y Robert Dale[9] han propuesto la forma más usual de agrupar las tareas de un sistema GLN en tres módulos básicos: *planeación de documento (document planning), generación de frases (microplanning) y realización de texto (sourface realisation)*. El módulo **planeación de documento** tiene dos tareas principales: *la determinación de contenido*, donde se decide qué se va a comunicar, y la *estructuración de documento,* que define cómo se va a comunicar la información. El módulo **generación de frases** tiene tres tareas: *la lexicalización*, escoge las palabras o recursos lingüísticos que deben usarse para expresar un contenido del dominio (ver Sección 3.3); *la agregación*, decide cómo agrupar estructuras lingüísticas (oraciones y párrafos) y la determinación de *expresiones referentes*, selecciona qué expresiones pueden ser usadas para referirse a entidades del dominio. El módulo **realización de texto** realiza dos tareas: *la realización lingüística,* que convierte las representaciones abstractas del dominio en texto real, y *la realización de estructura,* que convierte estructuras abstractas como párrafos y secciones en salidas como HTML, PDF, etc. [36, 7,10]

La forma como se conecten las tareas en un sistema GLN da como resultado una arquitectura (ver Figura 2).[29] Los sistemas Mumble[37], Text,[38, 39] Naos,[44, 45] y Wisber[26, 27] son representativos de una ***arquitectura secuencial o pipeline,***[8] ya que la información viaja a través de las tareas en una sola dirección. Kamp[2] es el ejemplo más conocido de un sistema con ***arquitectura integrada*** donde todas las decisiones son tomadas dentro de un proceso estructurado jerárquicamente pero no modularizado. Pauline[23, 24, 25] y Popel [14, 41, 49, 50] son sistemas construidos con ***arquitectura interactiva (feedback),*** donde se permite una revisión de las decisiones tomadas, ya que la información puede volver de nuevo a una tarea del sistema. El sistema Diógenes[42, 43] se enmarca dentro de una ***arquitectura blackboard*** donde los módulos ofrecen información sin conocer con exactitud cuál de los otros módulos la usará dentro de un mismo sitio de almacenamiento. Existen buenas propuestas de sistemas enmarcados en una ***arquitectura basada en revisión*** como KDS,[34] Yh,[16] Weiver[28]

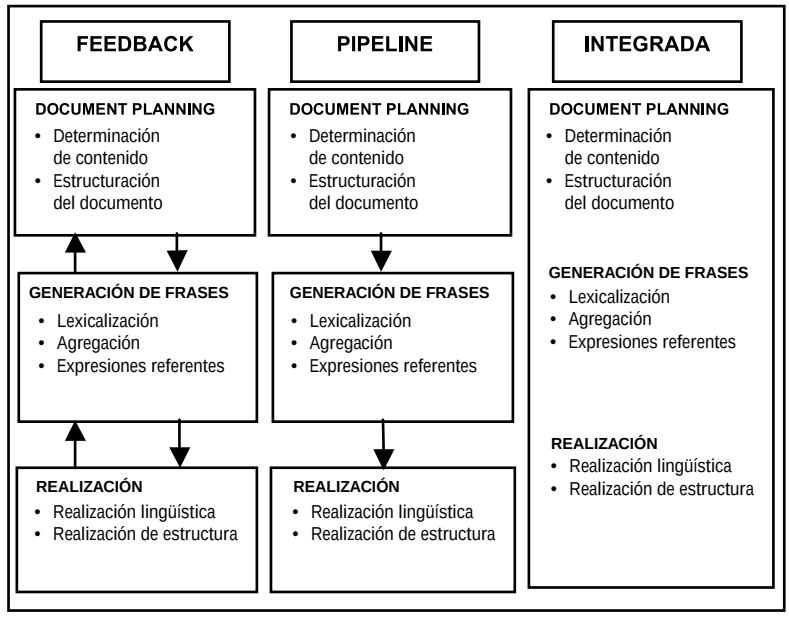

Figura 2. Arquitecturas en los sistemas GLN.

y Robin,[52] donde la información viaja por los módulos una y otra vez hasta obtener el resultado esperado.[29, 17, 19, 56, 40, 47]

En las siguientes secciones este artículo mostrará una propuesta que intenta independizar del dominio la construcción de un sistema GLN completo basado en patrones y características. La *sección 2* propone la construcción del sistema GLN en dos etapas, teniendo en cuenta la forma como Reiter y Dale proponen agrupar las tareas de estos sistemas; en la *sección 3* se propondrá una arquitectura para construir la segunda etapa del sistema GLN que soporta diferentes modelos de conexión entre los módulos y trata de independizar del dominio algunas tareas del sistema; finalmente la *sección 4* muestra conclusiones y recomendaciones relevantes acerca del trabajo realizado. La solución propuesta pretende: facilitar la construcción de sistemas completos de GLN, flexibilidad en la generación de frases, la inclusión de nuevos dominios de manera ortogonal, generación de textos en diversos formatos tales como texto sencillo, código HTML o formato PDF y en idiomas con lenguas romances como el español.

2. ETAPAS PARA LA CONSTRUCCIÓN DE SISTEMAS GLN COMPLETOS

La propuesta divide la construcción de un sistema GLN completo en dos etapas: *Etapa de Planificación del Contenido (EPC)* y *Etapa de Estructuración del Documento (EED)* (ver Figura 3). Se propone esta división debido a que las tareas de la etapa EPC dependen totalmente de la información que contendrá el texto a

generar mientras que las tareas de la etapa EED no, por lo que se puede diseñar una solución flexible que permita construir sistemas GLN extensibles a otros dominios (ver Sección 3). Estas etapas conectadas en forma secuencial formarán un sistema GLN completo, en el cual la salida de EPC será la entrada de EED.

2.1. Etapa de Planificación del Contenido (EPC)

Tiene como propósito dar estructura y orden al texto: escoger, agrupar y relacionar la información que aparecerá en el documento de salida. Para lograr estos objetivos el módulo EPC está dividido en dos submódulos: *determinación de contenido* y *estructuración de contenido*.

Antes de construir este módulo (implementación de la etapa) se debe realizar un análisis de requerimientos o *análisis de corpus*,[7, 54] que consiste en analizar las entradas y las salidas de los textos que se espera el sistema construya y con base en ese análisis se decide(n) la(s) meta(s) comunicativa(s), la información con la que trabajará el submódulo de Determinación de Contenido y la forma en que se agrupará la información a comunicar.

2.1.1 *Submódulo de Determinación de Contenido*

Es responsable de seleccionar la información que aparecerá en el texto de salida y estructurarla en mensajes,[9] que son elementos básicos o paquetes de información que el sistema de GLN manipulará. La información se encuentra en una *fuente de conocimiento* que típicamente está codificada en bases de datos y/o en bases de conocimiento, y es seleccionada según la *meta de comunicación* que es la que le informa al submódulo cuál es el objetivo del texto. Después de realizar el proceso de selección, la información deberá transformarse a una representación llamada mensajes.

2.1.2 *Submódulo de Estructuración de Contenido*

Este submódulo tiene como objetivo agrupar los mensajes generados en la Determinación de Contenido en

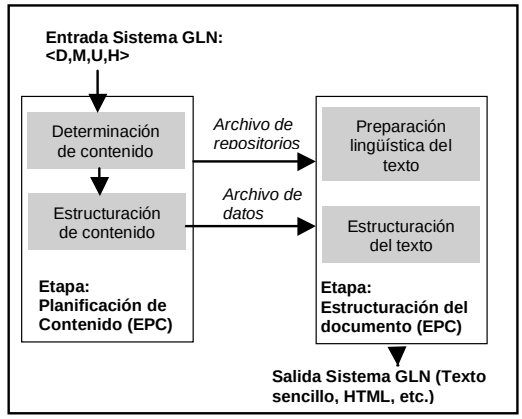

Figura 3. Etapas para un sistema GLN completo.

una representación abstracta del texto, ya sea utilizando *esquemas* o *Teoría de Estructura Retórica (TER).* Los esquemas son patrones que se diseñan para indicar el orden en que los mensajes serán agrupados.[9] Por su parte, una teoría de estructura retórica,[3, 58, 12] describe la organización de los textos en términos de relaciones de fondo, contraste, causa y efecto, etc. El resultado de agrupar los mensajes con cualquiera de los dos métodos mencionados anteriormente dará como resultado una representación abstracta del texto de forma jerárquica en el que los mensajes se encuentran en los niveles inferiores de la jerarquía. La propuesta aquí mencionada emplea la representación abstracta del texto mediante *TER,* debido a que se obtiene una mayor flexibilidad para organizar los mensajes que con *esquemas.*[9]

2.2. Etapa de Estructuración de Documento (EED)

Tiene como objetivo refinar la representación abstracta del texto que se formó en la etapa (EPC) a partir de la selección de estructuras lingüísticas y sintácticas adecuadas (*preparación lingüística del texto*); y obtener una secuencia de palabras, signos de puntuación y formatos de representación del texto (*realización del texto*).

Los esfuerzos de la propuesta de solución descrita en este artículo se centran en la construcción del módulo EED debido a que se quiere lograr un alto grado de independencia del dominio. En la siguiente sección se detalla una arquitectura para la etapa EED que facilita la construcción de sistemas completos de GLN y permite construir sistemas de GLN basados en patrones y características, gracias a las bondades de una gramática que intenta modelar la construcción de un texto.

3. ARQUITECTURA DEL MÓDULO EED

Esta arquitectura pretende facilitar la construcción de sistemas completos de GLN y soportar diferentes modelos de conexión (pipeline, feedback, etc.), y permite que los objetivos de la *etapa de estructuración de documento* se realicen de manera automática y en lo posible independiente del dominio del sistema GLN.

Como se observa en la Figura 4, la arquitectura está compuesta por tres componentes básicos: *controlador*, *preparación lingüística del texto* y *realización del texto*, y tiene tres entradas: el ***archivo de datos,*** que será construido por el submódulo de estructuración de contenido a partir de la *representación abstracta del texto* usando la gramática que se muestra en la Figura 5 y las características de la Figura 6; el ***archivo de repositorios*** (ver Figura 7) que contendrá la información que se almacena en los repositorios de datos, y el ***programa control,*** que tiene como objetivo indicarle al controlador el orden en que los módulos desempeñarán su función.

Antes de realizar el proceso de GLN con esta arquitectura, el controlador debe alimentar los *repositorios de datos* de los submódulos con la información del archivo de repositorios. Después de esta primera tarea el controlador ejecutará las instrucciones del *programa control* para obtener como salida un texto generado automáticamente.

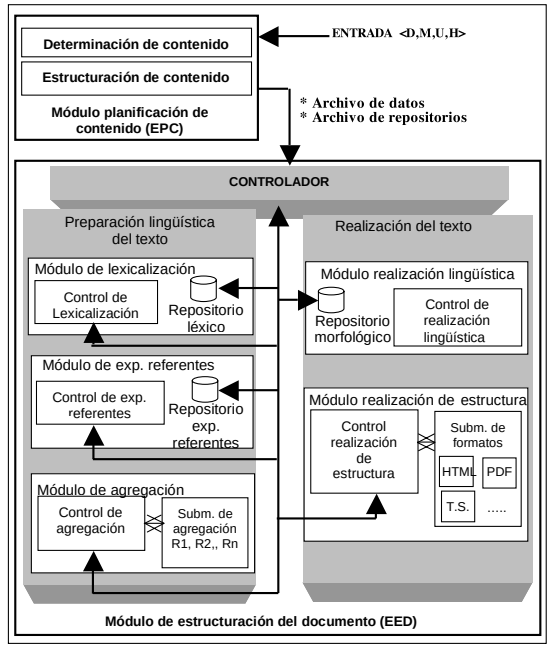

Figura 4: Arquitectura del módulo EED

3.1. Entradas

3.1.1 *Archivo de Datos*

Esta arquitectura debe permitir la construcción de sistemas de GLN para diferentes dominios. Esta información se puede modelar mediante plantillas o patrones y características. Las plantillas, aunque son usadas comúnmente, no brindan suficiente flexibilidad, ya que ajustar el sistema a un nuevo dominio puede implicar la creación de nuevas plantillas y/o la reconstrucción del sistema completo; por su parte, un sistema basado en patrones y características permite más flexibilidad a la hora de ajustar el sistema GLN a nuevos dominios, por lo que se seleccionó esta forma de modelar la información. Para ello se usa la gramática del lenguaje español[55] que se observa en la Figura 5, aumentada con los conceptos de texto, título y párrafo.

Esta representación permite organizar los mensajes de forma jerárquica y obtener un texto en diferentes formatos (HTML, PDF, Texto Sencillo, etc.). Adicionalmente existe un conjunto de características para describir los mensajes y facilitar la conjugación de verbos, y el uso de los artículos y pronombres (ver Figura 6). Una ventaja adicional de esta representación es que permite, para idiomas diferentes del español, ajustar los patrones y las características.

3.1.2 *Archivo de Repositorios*

Los repositorios de datos en los módulos son responsables de almacenar información propia del dominio

texto : título párrafo texto
j título tiporst párrafo restpárrafo
j título párrafo
j párrafo texto
j párrafo
j tiporst párrafo restpárrafo
j oración
título : oración j sujeto
párrafo : tiporst oración restoración
j oración restoración
j título párrafo
j párrafo
restpárrafo : tiporst párrafo restpárrafo
j párrafo restpárrafo
j párrafo
oración : tiporación sujeto VERBO predicado
restoración : tiporst oración restoración
j oración restoración
j oración
sujeto : SUSTANTIVO sujeto
j ARTÍCULO sujeto
j PRONOMBRE sujeto
j ADJETIVO sujeto
j COMNOMINAL sujeto
j SUSTANTIVO
j PRONOMBRE
j ADJETIVO
predicado : ADVERBIO predicado
j VERBO predicado
j tipocom sujeto predicado
j tipocom sujeto
j ADVERBIO
j VERBO

tiporst : AUMENTO
j CAUSA-EFECTO
j ÉNFASIS
j CONTRASTE
j CONDICIÓN
j RESUMEN
j CONJUNCIÓN
j DISJUNCIÓN
j EJEMPLO
tiporación : NEGATIVA
j AFIRMATIVA
j COMPARATIVA
j INTMANERA
j INTCANTIDAD
j INTTIEMPO
j INTRAZON
j INTSELECCIÓN
j INTLUGAR
j INTPERSONA
tipocom : COMNOMINAL
j COMDIRECTO
j COMINDIRECTO
j COMCIRCUNSTANCIAL

Figura 5: Gramática o patrones para construir un texto

Sustantivo = género, número, clase, tipo, lenguaje
Adjetivo = género, número, clase, tipo, lenguaje
Pronombre = género, número, clase, tipo, lenguaje
Artículo = género, número, clase, lenguaje
Preposición = se ubican según el tipo de complemento (ver en Figura 5 tipocom)
Verbo = número, persona, forma, tiempo, modo, claseverbo, voz, lenguaje

Figura 6. Características para describir los mensajes.

diferente de los mensajes y son los únicos elementos de la arquitectura del módulo EED dependientes del dominio del sistema de GLN a construir. La información contenida en los repositorios es almacenada por el controlador a partir del archivo de repositorios, que como se observa en la Figura 7 está dividido en tres secciones principales, ***lexicalizar***, ***conjugaciones*** y ***referencias,*** para alimentar el *repositorio léxico*, el *repositorio morfológico* y el *repositorio expresiones referentes* respectivamente. La sección **lexicalizar** agrupa un conjunto de expresiones léxicas

(identificadas con la etiqueta *dato*) que expresan un mismo concepto. La selección de cuál expresión usar se realiza en el módulo de lexicalización, la sección **conjugaciones** se usa para que el módulo de realización lingüística construya de manera incremental los verbos del repositorio morfológico y la sección **referencias** le indica al módulo de expresiones referentes las diferentes formas de hacer referencia a una misma entidad. Orlando Correa, en el ejemplo de la Figura 7, puede ser referenciado como Señor o como Ingeniero. Además, el Archivo de Repositorios puede contener las secciones necesarias de acuerdo con el objetivo deseado.

3.1.3 *Programa Control*

Permite que el desarrollador del sistema GLN decida el orden en que los módulos de la *preparación lingüística* refinarán el archivo de datos, en qué orden la *realización del texto* convertirá el archivo de datos en un texto gramatical y sintácticamente correcto y cuándo generar la salida del sistema GLN y en qué formato, como se observa en la Figura 4.

3.2. Controlador

Es el corazón de la arquitectura, responsable de almacenar la información del *Archivo de Repositorios* en los repositorios de los módulos, de establecer cómo se interconectan los módulos de la arquitectura según como lo indique el *programa control* y de permitir la comunicación entre los módulos por medio del archivo de datos. Este componente brinda flexibilidad al desarrollador para construir sistemas con diferentes modelos de

```
<repositorios>
  <lexicalizar>
    <valor>
      <id>tiempo</id>
      <expresion>
        <dato>Este mes</dato>
        <dato>Enero</dato>
      </expresion>
    </valor>
  </lexicalizar>
  <conjugacion>
    <valor>
      <id>ocurrir</id>
      <expresion>
        <verbo modo="impersonal" clase="in¯nitivo" ... >ocurrir</verbo>
        <verbo modo="impersonal" clase="gerundio" ...>ocurriendo</verbo>
        <verbo modo="impersonal" clase="participio" ...>ocurrió</verbo>
      </expresion>
    </valor>
  </conjugacion>
  <referencias>
    <valor>
      <id>Orlando Correa</id>
      <expresion>
        <dato>Ingeniero</dato>
        <dato>Señor</dato>
      </expresion>
    </valor>
  </referencias>
</repositorios>
```

Figura 7. Ejemplo en XML de un Archivo de Repositorios.

conexión (secuencial, feedback, etc.) que permite decidir cómo el archivo de datos será refinado por los módulos hasta obtener la salida.

3.3. Módulos y submódulos

Son cajas negras en la arquitectura. Su objetivo es aplicar las técnicas de la lingüística computacional y de inteligencia artificial para refinar la representación abstracta del texto (Archivo de Datos) en un texto legible por el usuario y que sea gramaticalmente y sintácticamente correcto.

Módulo de Lexicalización. Es responsable de decidir qué palabras o recursos lingüísticos deben usarse para expresar un mensaje por medio de un *control de lexicalización* donde están los algoritmos encargados de realizar las escogencias léxicas y un *repositorio de lexicalización* que almacena las diferentes expresiones léxicas necesarias para expresar un mensaje dentro del documento. Por ejemplo una fecha se puede escribir: a) 24 de diciembre del 2006, b) 24 de diciembre del año en curso, c) Navidad, d) día 24 del último mes del presente año, etc. Estas diferentes formas de expresión son almacenadas en el repositorio y el control de lexicalización selecciona una opción por medio de redes de discriminación, árboles de decisión, heurísticos o aleatoriamente.[13, 20, 32, 1]

Módulo de Agregación. Por medio del *control de agregación* es responsable de combinar frases simples para formar oraciones complejas, usando reglas de agregación deducidas a partir de la teoría de estructura retórica (si dos mensajes están en una relación de *secuencia* ellos pueden ser unidos formando una oración) almacenadas en el *submódulo de reglas de agregación* y operaciones básicas sobre oraciones tales como una conjunción simple (O1 y O2), participantes compartidos (O1: Pepe juega. O2: Juanita juega, entonces O: Pepe y Juanita juegan) o construcción sintáctica (O1: Juan es doctor. O2: Juan trabaja. O: Juan trabaja como doctor). La agregación también puede ocurrir para construir párrafos. Sin embargo, las operaciones en párrafos son más elaboradas que con las oraciones.[9, 20, 21, 22]

Módulo de Expresiones Referentes. Es responsable de decidir qué expresión puede ser usada para referirse a una entidad del dominio conservando la intención del texto. La expresión común de referencia es el pronombre. Este módulo está formado por un *control de expresiones referentes,* donde están los algoritmos responsables de seleccionar la mejor referencia a una entidad y un *repositorio de expresiones referentes* que almacena entidades, como por ejemplo Estudiante y Señorita para referirse a una persona en particular.[21, 51, 6, 31, 22, 48]

Módulo de Realización Lingüística. Está encargado de convertir la representación abstracta del texto en secuencia de palabras para producir texto sintáctica y morfológicamente bien escrito, aplicando reglas gramaticales tales como: *el artículo precede solo al sustantivo*. Este módulo está compuesto por un *control de realización lingüística* donde están los algoritmos encargados de cumplir con el objetivo de este módulo y un *repositorio morfológico* donde el controlador almacena la conjugación de verbos diferentes del

ser, estar y haber (que por defecto están almacenados en el repositorio) a partir del archivo de repositorios. Adicionalmente este repositorio tiene almacenados pronombres, artículos y adjetivos que soportan el objetivo del módulo.[9, 33]

Módulo de Realización de Estructura. Este módulo está encargado de convertir estructuras abstractas como párrafos, secciones, títulos, etc., a un formato especial como HTML, PDF, LATEX, etc., por medio de un *control de realización de estructura* que se encarga de comunicarse con un *submódulo de formatos* para lograr el objetivo de este módulo.[9]

Los submódulos de la arquitectura serán cajas blancas en la medida en que es posible ampliarlos o desarrollar uno nuevo.

4. CONCLUSIONES Y TRABAJOS FUTUROS

La división del proceso de GLN presentada en este artículo (Sección 2), le permitirá al desarrollador construir un sistema GLN completo, concentrándose en el análisis de corpus y desarrollo de la etapa EPC para ser conectada con la etapa EED.

La arquitectura propuesta para la etapa EED (Sección 3) facilita la construcción de sistemas GLN completos, basados en patrones y características, y la reutilización del sistema para dominios diferentes y posiblemente más complejos permite la generación de texto en lenguajes romances, y flexibilidad en la generación de frases y textos en diferentes formatos.

Los módulos que componen la arquitectura de la etapa (EED) son independientes el uno del otro, lo que permite trabajar ortogonalmente en técnicas y metodologías particulares para cada módulo, de modo que se pueda avanzar de manera más rápida en el desarrollo de sistemas de GLN complejos y robustos.

A partir de la propuesta presentada en este artículo se refinará la gramática (patrones y características) para que reconozca oraciones diferentes de las oraciones simples, se trabajará en la generación de texto en idiomas diferentes del español, en el desarrollo de nuevas técnicas y de algoritmos robustos que realicen escogencias léxicas, teniendo en cuenta la meta de comunicación del texto, reglas de agregación genéricas que se ajusten a múltiples dominios aprovechando la independencia de los módulos. Finalmente se desarrollará una interfaz que facilite al desarrollador del sistema GLN almacenar la información escogida por el módulo (EPC) en los Archivos de Repositorios y de Datos.

BIBLIOGRAFÍA

1. A. Polguére. *A "natural" lexicalization model for Language Generation,* In Proceedings of the Fourth Symposium on Natural Language Processing (SNLP2000). Chiangmai, Thailand, 10-12 May 2000, pp. 37-50. 2000.
2. Appelt, D. *Planning English sentences*. Cambridge: Cambridge University Press. 1985.
3. Mann Bill. *An Introduction to Rhetorical Structure Theory (RST),* http://www.sil.org/linguistics/rst/rintro99.htm. 1999.

4. Appelt D. *Bidirectional grammars and the design of natural language generation systems*. Theoretical Issues in Natural Language Processing - 3, New Mexico, 185-191. 1987.
5. Hovy E.H. Language Generation, *Survey of the State of the Art in Human Language Technology*, 1996.
6. Krahmer E. S. van Erk and A. Verleg. *A Meta-Algorithm for the Generation of Referring Expressions*. In: Proceedings of the 8th European Workshop on Natural Language Generation, Toulouse, 2001.
7. Reiter E. S Sripada, and R Robertson. *Acquiring Correct Knowledge for Natural Language Generation*. Journal of Artificial Intelligence Research 18. 491-516. 2003.
8. Reiter E. *Pipelines and size constraints*. Computational Linguistics. Forthcoming. 2000a.
9. Reiter E. and Robert Dale. *Building Natural Language Generation Systems*. Cambridge University. 2000b.
10. Reiter E. and Robert Dale. *Tutorial on Building Applied Natural Language Generation Systems*. Cambridge University Press. 1997.
11. Reiter E. C. Mellish, and J. Levine. *Automatic generation of technical documentation*. Applied Artificial Intelligence, 9, 1995.
12. Forsbom Eva *Rhetorical Structure Theory in Natural Language Generation*. Uppsala University and GSLT. 2005.
13. Benamara Farah, Patrick Saint-Dizier. *Lexicalization Strategies in Cooperative Question-Answering Systems*. MIT Press, p. 345-352. Dans: COLING 04, Genµeve, 22 aout 26 aout 2004.
14. Finkler W, Neumann G. Popel-How: A distributed parallel model for incremental natural language production with feedback. Proceedings of the 11th International Joint Conference on Artificial Intelligence, Detroit (pp. 1518-1523). 1989.
15. Fundación Duques de Soria. Conversar con el ordenador: el procesamiento del lenguaje y del habla en los sistemas de diálogo. Curso de Industrias de la Lengua, julio 2003.
16. R. Gabriel Deliberate writing. In D. McDonald and L. Bolc (Eds.). Natural language generation systems (pp. 1-46). Berlin: Springer. 1986.
17. García C., Hervás, R., Gervás, P. Una arquitectura software para el desarrollo de aplicaciones de generación de lenguaje natural. Sociedad Española para el Procesamiento del Lenguaje Natural, Procesamiento de Lenguaje Natural, No. 33, septiembre de 2004, ISSN: 1135-5948.
18. Graham Wilcock. Pipelines, templates and transformations: XML for natural language generation. In Proceedings of the 1st NLP and XML Workshop, pages 1-8, Tokyo, 2001.
19. Hervás R., Gervás, P. Uso flexible de soluciones evolutivas para tareas de Generación de Lenguaje Natural. Sociedad Española para el Procesamiento del Lenguaje Natural, Procesamiento de Lenguaje Natural, No. 35, septiembre de 2005a, ISSN: 1135-5948.

20. Hervás R., Gervás, P. Case Retrieval Nets for Heuristic Lexicalization in Natural Language Generation. 12th Portuguese Conference on Artificial Intelligence (EPIA 05), Coimbra, Portugal, Springer LNAI Series, 2005b.
21. Hervás R., Gervás, P. An Evolutionary Approach to Referring Expression Generation and Aggregation. (Poster) 10th EuropeanWorkshop on Natural Language Generation, Aberdeen, Scotland, 8-10 August 2005c.
22. Hervás R., Gervás, P. Applying Genetic Algorithms to Referring Expression Generation. Tenth International Conference on Computer Aided Systems Theory, (EUROCAST2005), Las Palmas de Gran Canaria, Spain, February 7-11, 2005d.
23. Hovy, E. Planning coherent multisentential text. Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics, Bufalo, NY (pp. 163-169). 1988a.
24. Hovy, E. Generating natural language under pragmatic constraints. Hillsdale, NJ: Earlbaum. 1988b.
25 Hovy, E. Pragmatics and natural language generation. Artificial Intelligence, 43, 153-197. 1990.
26. Horacek, H. The architecture of a generation component in a complete natural language system. In R. Dale, C. Mellish and M. Zock (Eds.), Current research in natural language generation (pp. 193-227). London: Academic Press. 1990.
27. Horacek H. and Pyka C. Towards bridging two levels of representation: Linking the syntactic-functional and object-oriented paradigms. In J.-L. Lassez and F. Chin (Eds.), International Computer Science Conference 88 Artificial Intelligence: Theory and Applications, Hong Kong (pp.281-288). 1988.
28. Inui K., Tokunaga T., Tanaka H. Text revision: A model and its implementation. In R. Dale, E. Hovy, D. RÄosner and O. Stock (Eds.), Aspects of automated natural language generation (pp. 215-230). Berlin: Springer. 1992
29. Koenraad De Smedt, Helmut Horacek, and Michael Zock. Architectures for Natural Language Generation: Problems and Perspectives. In Giovanni Adorni and Michael Zock, editors, Trends in Natural Language Generation: An Articial Intelligence Perspective; Fourth European Workshop, EWNLG '93, Pisa, Italy, April 1993.
30. K. VAN Deemter, E. Kramer, and M. Theune. Plan-based vs. template-based NLG: a false opposition? In Proceedings of the Workshop on Natural Language Systems at the German Annual Conference on Artificial Intelligence, Bonn, Germany, September 13-15, 1999.
31. K. VAN Deemter and Magn¶us M. Halldórsson. Logical Form Equivalence: the case of Referring Expressions Generation, in Procs. of 8th Europaean Workshop on Natural Language Generation (EWNLG2001), Toulouse. 2001.
32. Makoto Kanazawa and Ryo Yoshinaka. Lexicalization of second-order ACGs. NII Technical Report. NII-2005-012E. National

Institute of Informatics, Tokyo. 2005.
33. Manfred Stede. Non-Parametric Statistics for the Behavioural Sciences. McGraw-Hill, New York. A generative perspective on verb alternations. Computational Linguistics, 24(3):401-430. 1998.
34. MANN W, Moore J. Computer generation of multiparagraph English text. American Journal of Computational Linguistics, 7, 17-29. 1981
35. Mariët Theune. From Monologue to Dialogue: Natural Language Generation in OVIS. AAAI 2003 Spring Symposium on Natural Language Generation in Written and Spoken Dialogue, Palo Alto, USA, pages 141-150.
36. Haase Martin. Aspects of Natural Language Generation and Prosody. 2001.
37. McDonald, D. Natural language generation as a computational problem: An introduction. In M. Brady and R. Berwick (Eds.), Computational models of discourse (pp. 209-266). Cambridge, MA: MIT Press. 1983.
38. McKeown, K. The Text system for natural language generation: An overview. Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics, Toronto (pp. 113-120). 1982.
39. McKeown, K. Text generation using discourse strategies and focus constraints to generate natural language text. Cambridge: Cambridge University Press. 1985.
40. White Michael and Ted Caldwell. Examplars: A practical, extensible framework for dynamic text generation. In Proceedings of the 9th International Workshop on Natural Language Generation, pages 266{275, Niagara-on-the-Lake, Ontario, 1998.
41. Neumann G, Finkler W. A head-driven approach to incremental and parallel generation of syntactic structures. Proceedings of the 13th International Conference on Computacional Linguistics, Helsinki (Vol. 2, pp. 288-293). 1990.
42. Nirenburg S, Nirenburg I. A framework for lexical selection in natural language generation. Proceedings of the 12th International Conference on Computational Linguistics, Budapest (pp. 471-475). 1988.
43. Nirenburg S, Lesser V, Nyberg E. Controlling a language generation planner. Proceedings of the 11th International Joint Conference on Arti¯cial Intelligence (pp. 1524-1530). 1989
44. Novak, H.-J. Strategies for generating coherent descriptions of object movements in street scenes. In G. Kempen (Ed.), Natural language generation: New results in artificial intelligence, psychology and linguistics (pp. 117?132). Dordrecht: Nijhoff (Kluwer). 1987a.
45. Novak, H.-J. Textgenerierung von visuellen Daten: Beschreibungen von StraBenszenen. Berlin: Springer. 1987b.
46. P. Gervas, Un modelo computacional para la generación automática de poesía formal en castellano, Procesamiento de lenguaje natural, 26(26). 2000a.
47. P. Heeman, G. Hirst. Collaborating on referring expressions. In:

Computational Linguistics 21 (3), 1995.

48. Reithinger N. Popel: A parallel and incremental natural language generation system. In C.L. Paris, W.R. Swartout and W.C. Mann (Eds.), Natural language generation in artificial intelligence and computational linguistics (pp. 179-199). Boston: Kluwer Academia Publishers. 1991.
49. Reithinger N. The performance of an incremental generation component in multi-modal dialog contributions. In R. Dale, E. Hovy, D. RÄosner and O. Stock (Eds.), Aspects of automated natural language generation (pp. 263-276). Berlin: Springer. 1992.
50. Dale Robert and Ehud Reiter. Computational interpretation of the Gricean maxims in the generation of referring expressions. Cognitive Science. 1995. 19(8):233-263.
51. Robin J. A revision-based generation architecture for reporting facts in their historical context. In H. Horacek and M. Zock (Eds.), New concepts in natural language generation: Planning, realization, and systems (pp. 238-268). London: Pinter. 1993.
52. S. Bangalore, O. Rambow, and M. Walker. Natural language generation in dialog systems. In Proceedings of Human Language Technology Conference, 2003.
53. Sabine Geldof. Corpus-analysis for NLG. In: Proceedings of the European Workshop on Natural Language Generation (ENLG'02) Budapest, Hungary. Abril 2003.
54. Revilla Santiago. Gramatica Espa~nola Moderna Teoria y Ejercicios. McGraw-Hill. 1975.
55. Seneff Stephanie and Joseph Polifroni. Formal and natural language generation in the Mercury conversational system. In Proceedings of the Sixth International Conference on Spoken Language Processing (ICSLP2000), October 2000.
56. Salazar William Ángel. Alta Redacción. Informes Técnicos y Administrativos. 2004.
57. William C. Mann and Sandra A. Thompson. Rhetorical Structure Theory: A theory of text organization. In Livia Polanyi, editor, The Structure of Discourse. pages 85{96. Ablex Publishing Company, Norwood, NJ, 1987.
58. Yohei Seki, Aoyama Gakuin and Kenichi Harada. Machine Translation based on NLG from XML-DB. In The 17th International Conference on Computational Linguistics. 2000.

CURRÍCULOS

Gloria Johanna Chala Torres. Analista de Sistemas, GC2 Carvajal S.A. Graduada de la Universidad Javeriana (Cali), 2006.

Rafael Armando Jordán. Profesor en el Departamento de Ingeniería de la Computación, Universidad Javeriana (Cali).

Diego Luis Linares. Ph.D. en Formas e Inteligencia Artificial por la Universidad Politécnica de Valencia, España. Coordinador de Investigación de la Universidad Javeriana (Cali).